

SISTEMAS BIOINSPIRADOS

MAGÍSTER EN INTELIGENCIA ARTIFICIAL

JULIO-AGOSTO 2022

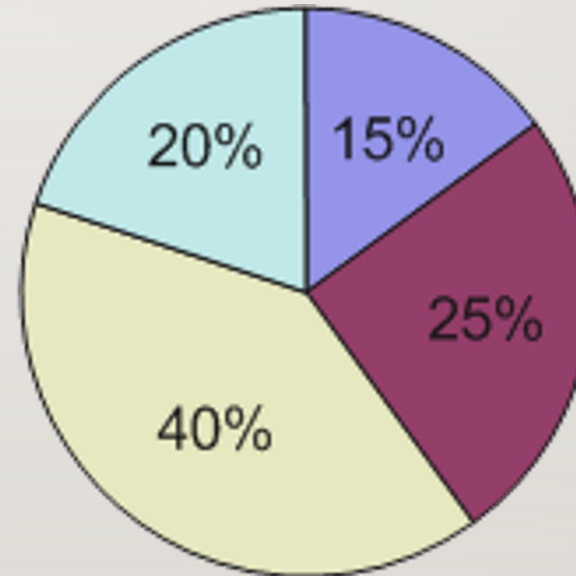
PROFESOR: RICARDO CONTRERAS A.



SELECCIÓN (USANDO RULETA)

- Distribución de probabilidad basada en el fitness

1	000110110	15%
2	111001101	25%
3	000110110	40%
4	111101111	20%



CONSTRUCCIÓN DE RULETA (VALORES DE FITNESS POSITIVOS)

- Calcular el valor del fitness $\text{eval}(v_i)$ para cada cromosoma v_i ($i = 1, \dots, \text{tam_pobl}$)
- Encontrar el fitness total de la población: $F = \sum \text{eval}(v_i)$ ($i = 1, \dots, \text{tam_pobl}$)
- Calcular la probabilidad de una selección p_i para cada cromosoma v_i : $p_i = \text{eval}(v_i)/F$
- Calcular la probabilidad acumulativa q_i para cada cromosoma v_i : $q_i = \sum p_j$ ($j = 1, \dots, i$)

EL PROCESO DE SELECCIÓN

- Se basa en el giro de la ruleta: se hace girar tam_pobl veces, seleccionando cada vez un solo cromosoma para una nueva población
- Se genera un número random r en $[0, 1]$
- Si $r < q_1$, entonces se selecciona el primer cromosoma, v_1 ; en caso contrario se selecciona el i -ésimo cromosoma v_i , tal que $q_{i-1} < r \leq q_i$

- Algunos cromosomas pueden ser seleccionados más de una vez
-

- Percepción de supervivencia:
 - Los mejores se reproducen más
 - Los casos “promedio” se mantienen
 - Los peores mueren

PROBABILIDAD DE CRUZAMIENTO P_c

- Esta probabilidad nos da el número esperado de cromosomas que participarán en la operación de cruzamiento: $p_c \cdot \text{tam_pobl}$
- Operación. Para cada cromosoma en la (nueva) población:
 - Genere un número random r en $[0,1]$
 - Si $r < p_c$, seleccione el cromosoma dado para cruzamiento

CRUZAMIENTO SIMPLE

- Se seleccionan aleatoriamente las parejas
- Para cada par de cromosomas emparejados, se genera un número random **pos** en el rango $[1, m-1]$
- El número **pos** indica la posición del punto de cruzamiento

EJEMPLO

- Dos cromosomas
 - $(b_1 b_2 \dots b_{pos} b_{pos+1} \dots b_m)$ y
 - $(c_1 c_2 \dots c_{pos} c_{pos+1} \dots c_m)$
- Son reemplazados por el par de descendientes
 - $(b_1 b_2 \dots b_{pos} c_{pos+1} \dots b_m)$ y
 - $(c_1 c_2 \dots c_{pos} b_{pos+1} \dots b_m)$

PROBABILIDAD DE MUTACIÓN P_M

- Nos da el número esperado de bits mutados $p_m \cdot m \cdot \text{tam_pobl}$
- Todos los bits (en cada cromosoma, en toda la población) tienen igual probabilidad de sufrir una mutación (cambiar un 0 por un 1 o vice-versa)

MUTACIÓN

- Por cada cromosoma en la población actual (o sea, después del cruzamiento) y para cada bit dentro del cromosoma:
 - Generar un número random r en el rango $[0,1]$
 - Si $r < p_m$, mutar el bit

ES UN PROCESO EVOLUTIVO

- Luego de la selección, el cruzamiento y la mutación, la nueva población está lista para su próxima evaluación
- Se construye nueva ruleta de acuerdo a los valores actuales de los fitness
- El resto de la evolución es, simplemente, la repetición cíclica de los pasos anteriores

¿POR QUÉ FUNCIONAN LOS AGS?

- Los fundamentos teóricos de los AGs descansan en la representación de soluciones sobre strings binarios y en la noción de esquema.
- Un esquema se construye por medio de la introducción de un símbolo **don't care** (*) en el alfabeto genético.

-
- Un esquema representa a todos los strings (un hiperplano o subconjunto del espacio de búsqueda) que coinciden en todas aquellas posiciones en que no hay un símbolo *
 - Por ejemplo, el esquema (*IIII00I00) parea dos strings:
(**I**IIII00I00) y (**0**IIII00I00)

- El “esquema” (1001110001) representa a un solo string (el mismo)
-
- El esquema (*****) representa a todos los strings de longitud 10
 - Si r es el número de símbolos **don't care**, cada esquemaanea exactamente 2^r strings
 - Cada string de longitud m es pareado por 2^m esquemas

PROPIEDADES DE UN ESQUEMA

- Orden
- Longitud de definición

ORDEN DE UN ESQUEMA

- Es el número de posiciones fijas (0 o 1) presentes en el esquema, y se denota $o(S)$
- O sea, es la longitud de la plantilla menos el número de símbolos **don't care**
- El orden define la especialidad de un esquema

LONGITUD DE DEFINICIÓN DEL ESQUEMA

- Es la distancia entre la primera y la última posición fija del esquema
- Define el grado de compactación de la información contenida en el esquema
- Se denota por $\delta(S)$

EJEMPLO

- $S_1 = (**001*110)$
- $S_2 = (****00**0*)$
- $S_3 = (11101**001)$

- $o(S_1) = 6, o(S_2) = 3, o(S_3) = 8;$

- $\delta(S_1) = 10 - 4 = 6, \delta(S_2) = 9 - 5 = 4, \delta(S_3) = 10 - 1 = 9$

-
- Cruzamiento y mutación pueden destruir y crear instancias de un esquema.
 - Un esquema S sobrevive al cruzamiento si uno de los descendientes también es una instancia de él.
 - La probabilidad de sobrevivir bajo **cruzamiento** es mayor para esquemas de longitud **más corta**.
 - La probabilidad de sobrevivir bajo **mutación** es mayor para esquemas de **orden bajo**.

EJEMPLO GENERAL

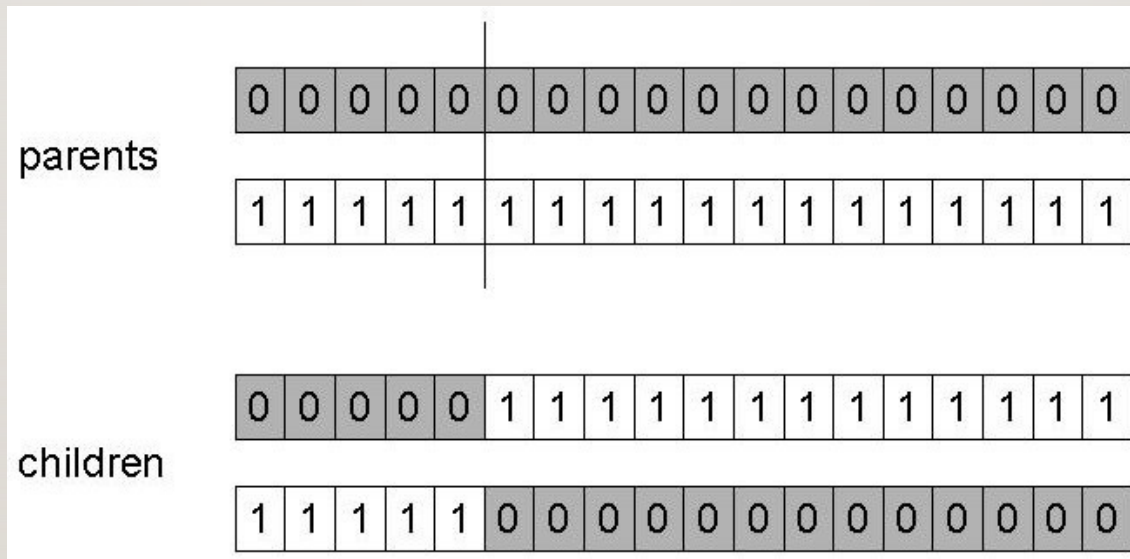
- Los pasos relevantes.
- Cualquier duda, me interrumpen

ALGORITMO GENÉTICO SIMPLE

1. Seleccionar padres para cruzar
(Miembros de la población candidatos a cruzamiento = todos)
2. Revolver (como en un juego de cartas)
3. Para cada par consecutivo aplique cruzamiento con probabilidad p_c , si no hay cruzamiento, copiar los padres
4. Para cada descendiente, aplique mutación (cambiar el bit con probabilidad p_m independiente para cada bit)
5. Reemplace toda la población con los *nuevos* individuos

OPERADORES: CRUZAMIENTO (SIMPLE)

- Elección de un punto random en el cromosoma
- Crear dos hijos de acuerdo a la división recién determinada
- P_c es un valor que típicamente está en el intervalo (0.6, 0.9)



OPERADORES: MUTACIÓN

- Alterar cada gen independientemente con una probabilidad p_m
- p_m se llama tasa de mutación
 - Típicamente entre $1/(\text{tamaño_población})$ y $1/(\text{largo_cromosoma})$

parent

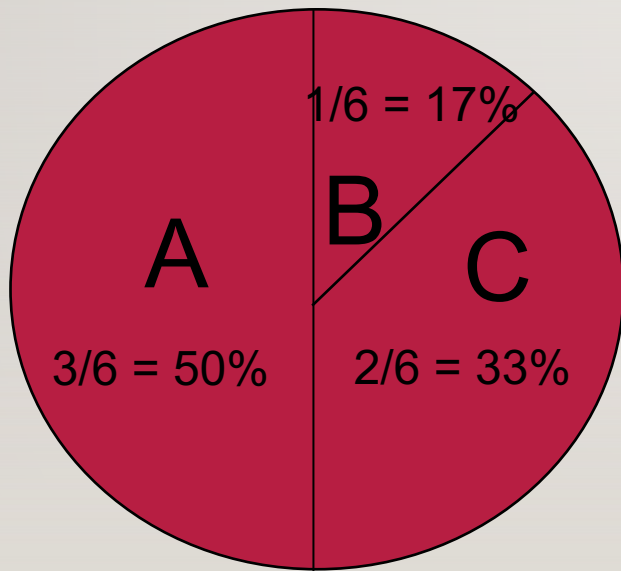
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

child

0	1	0	0	1	0	1	1	0	0	0	1	0	1	1	0	0	1
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

OPERADORES: SELECCIÓN

- Idea: los mejores individuos tienen más probabilidades
 - Proporcional al fitness
 - Implementación: técnica de la ruleta
 - Asignar a cada individuo una parte de la ruleta
 - Girar la ruleta n veces para seleccionar n individuos



fitness(A) = 3

fitness(B) = 1

fitness(C) = 2

EJEMPLO (GOLDBERG)

- Problema: maximizar x^2 sobre $\{0, 1, \dots, 31\}$
- Aproximación usando AGs:
 - Representación: codificación binaria, por ejemplo $01101 \leftrightarrow 13$
 - Tamaño de población: 4
 - Selección a través de ruleta
 - Inicialización: random
- Se muestra un ciclo generacional

SELECCIÓN

String no.	Initial population	x Value	Fitness $f(x) = x^2$	$Prob_i$	Expected count	Actual count
1	0 1 1 0 1	13	169	0.14	0.58	1
2	1 1 0 0 0	24	576	0.49	1.97	2
3	0 1 0 0 0	8	64	0.06	0.22	0
4	1 0 0 1 1	19	361	0.31	1.23	1
Sum			1170	1.00	4.00	4
Average			293	0.25	1.00	1
Max			576	0.49	1.97	2

CRUZAMIENTO

String no.	Mating pool	Crossover point	Offspring after xover	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 1	4	0 1 1 0 0	12	144
2	1 1 0 0 0	4	1 1 0 0 1	25	625
2	1 1 0 0 0	2	1 1 0 1 1	27	729
4	1 0 0 1 1	2	1 0 0 0 0	16	256
Sum					1754
Average					439
Max					729

MUTACIÓN

String no.	Offspring after xover	Offspring after mutation	x Value	Fitness $f(x) = x^2$
1	0 1 1 0 0	1 1 1 0 0	26	676
2	1 1 0 0 1	1 1 0 0 1	25	625
2	1 1 0 1 1	1 1 0 1 1	27	729
4	1 0 0 0 0	1 0 1 0 0	18	324
Sum				2354
Average				588.5
Max				729

ALGUNAS LIMITACIONES

- Representación es restringida
- Mutación & cruzamiento sólo es aplicable para strings de bits & representaciones enteras
- Mecanismos de selección son sensibles a las poblaciones convergentes con valores cercanos de fitness
- Se puede introducir mejoras

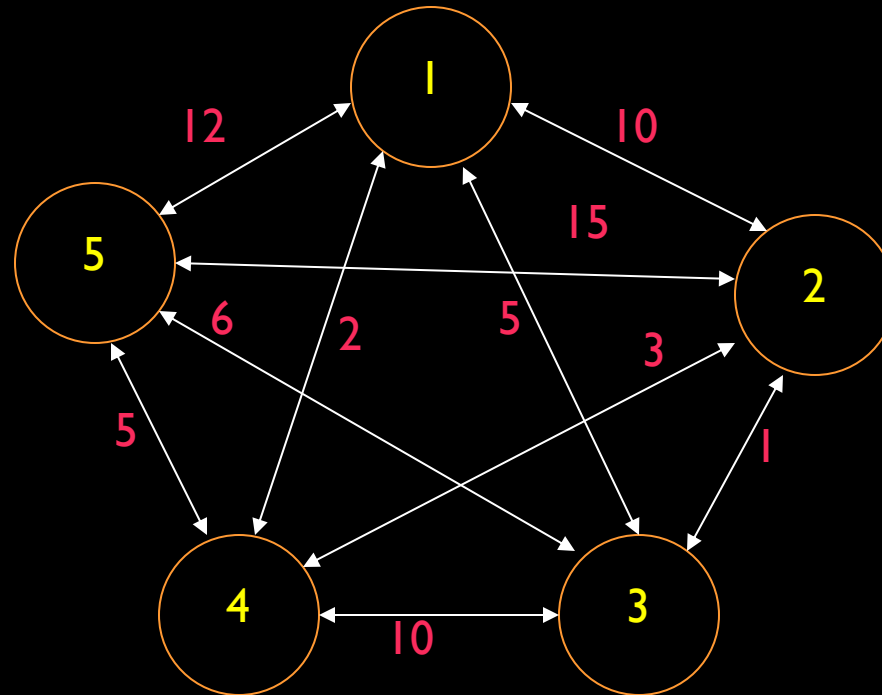
¿CÓMO RESOLVERLO?

- A través de operadores más especializados
 - Por ejemplo OX
 - Lo veremos pronto

PROBLEMA DEL VENDEDOR VIAJERO (TSP)

- El vendedor viajero debe visitar cada ciudad de su territorio exactamente una vez y luego regresar a su punto de partida.
- Dado el costo de viaje entre cada una de las ciudades, él debería planificar su itinerario de manera que el costo total sea mínimo.
- Este es un problema de optimización combinatoria que aparece en numerosas aplicaciones

EJEMPLO



REPRESENTACIÓN

- ¿deberíamos dejar que el cromosoma sea un vector de enteros?
- ¿deberíamos transformar cada entero a su equivalente en binario?
- ¿cuáles son las ventajas y problemas de cada una de estas representaciones?

REPRESENTACIÓN

- La representación más natural es el vector de enteros
- $v = (i_1 i_2 \dots i_m)$ representa un itinerario: desde i_1 a i_2 , etc, desde i_{m-1} a i_m y regresa a i_1
- v es una permutación de $(1 \ 2 \ \dots \ m)$
- ejemplos:
 - $v1 = (41523)$
 - $v2 = (13542)$
 - $v3 = (45213)$
 - $v4 = (35412)$

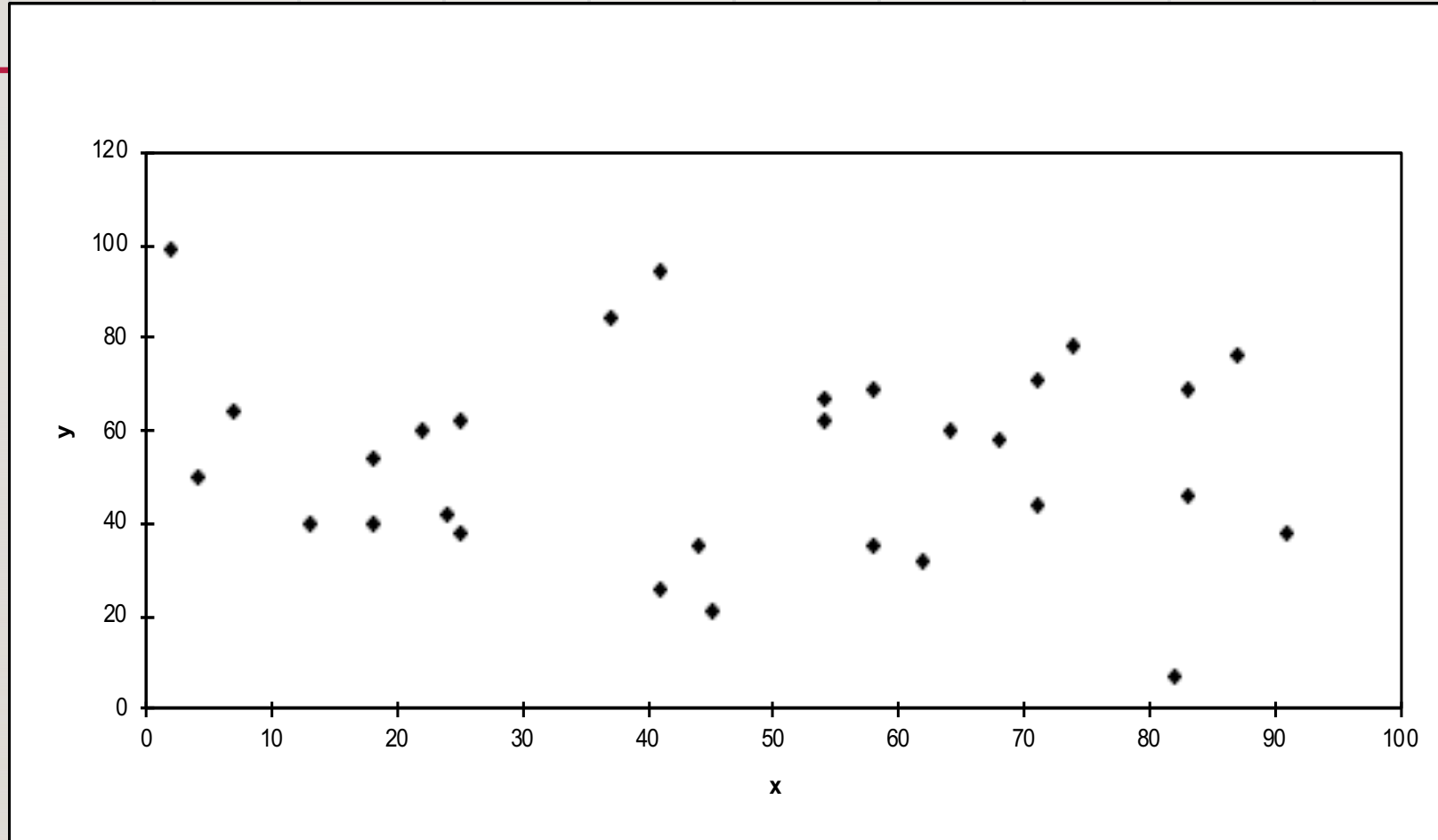
POBLACIÓN INICIAL

- Para el proceso de inicialización se puede usar alguna heurística
 - aceptar unas pocas salidas de algún algoritmo para el TSP, partiendo de diferentes ciudades
- Se puede inicializar la población creando un algoritmo que genere aleatoriamente permutaciones de $(1\ 2 \dots m)$

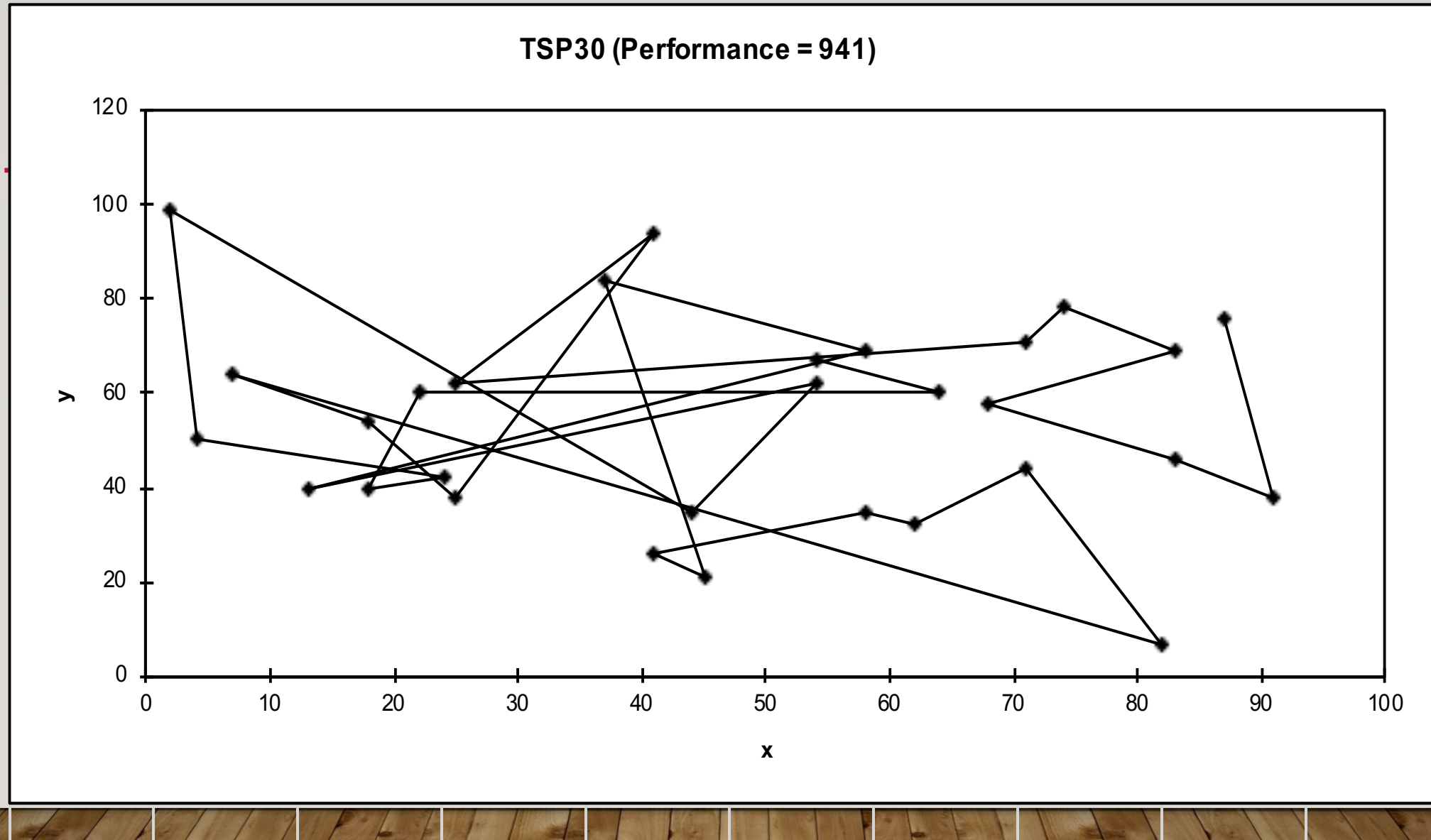
FUNCIÓN DE EVALUACIÓN

- La evaluación del cromosoma está dada por la suma de los costos de viajar entre las ciudades
- ejemplo
 - $v1 = (41523)$ $f(v1) = 2 + 12 + 15 + 1 + 10 = 40$
 - $v2 = (13542)$ $f(v2) = 5 + 6 + 5 + 3 + 10 = 29$
 - $v3 = (45213)$ $f(v3) = 5 + 15 + 10 + 5 + 10 = 45$
 - $v4 = (35412)$ $f(v4) = 6 + 5 + 2 + 10 + 1 = 24$

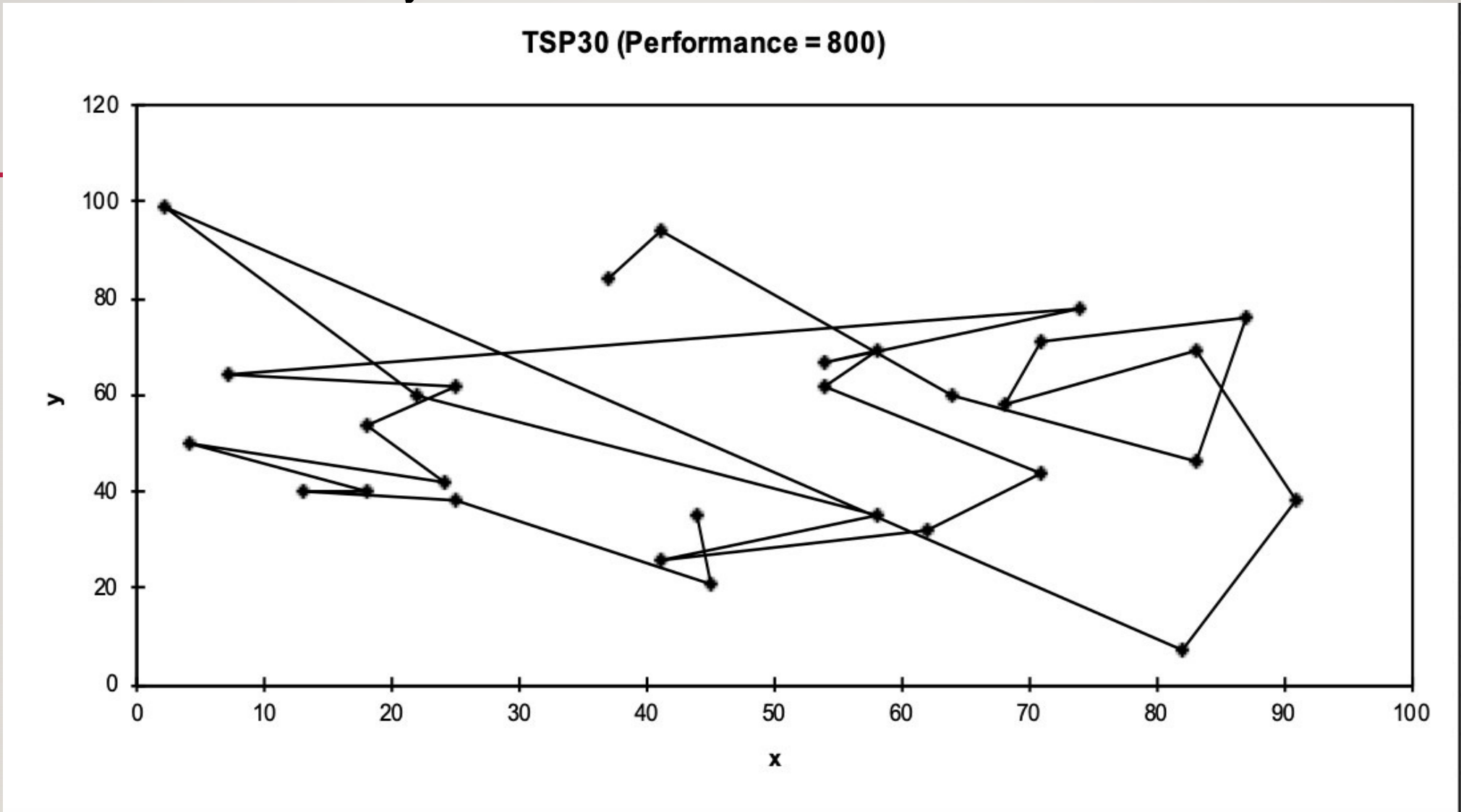
TSP EJEMPLO: 30 CIUDADES



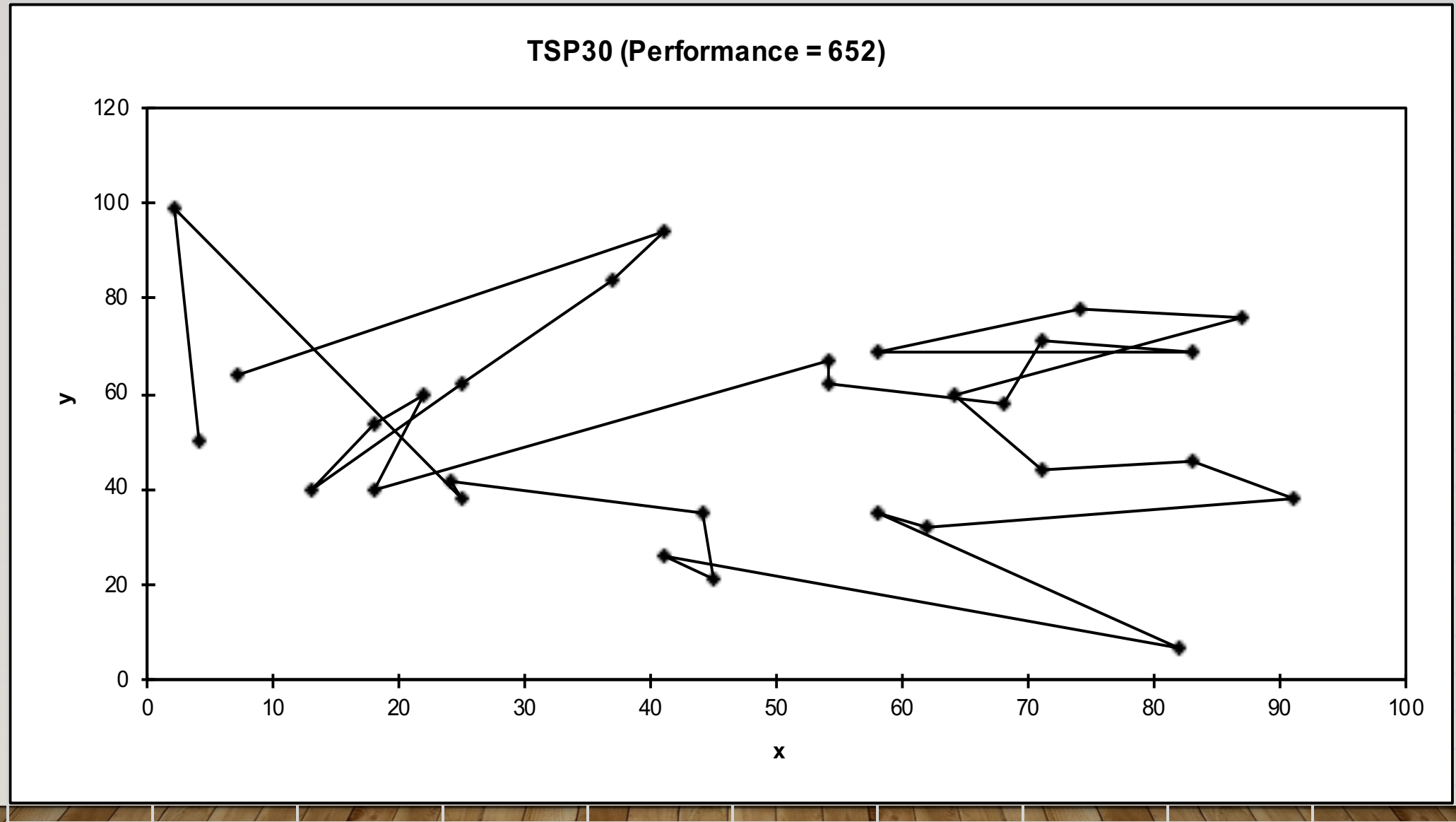
SOLUCION , (DISTANCIA = 941)



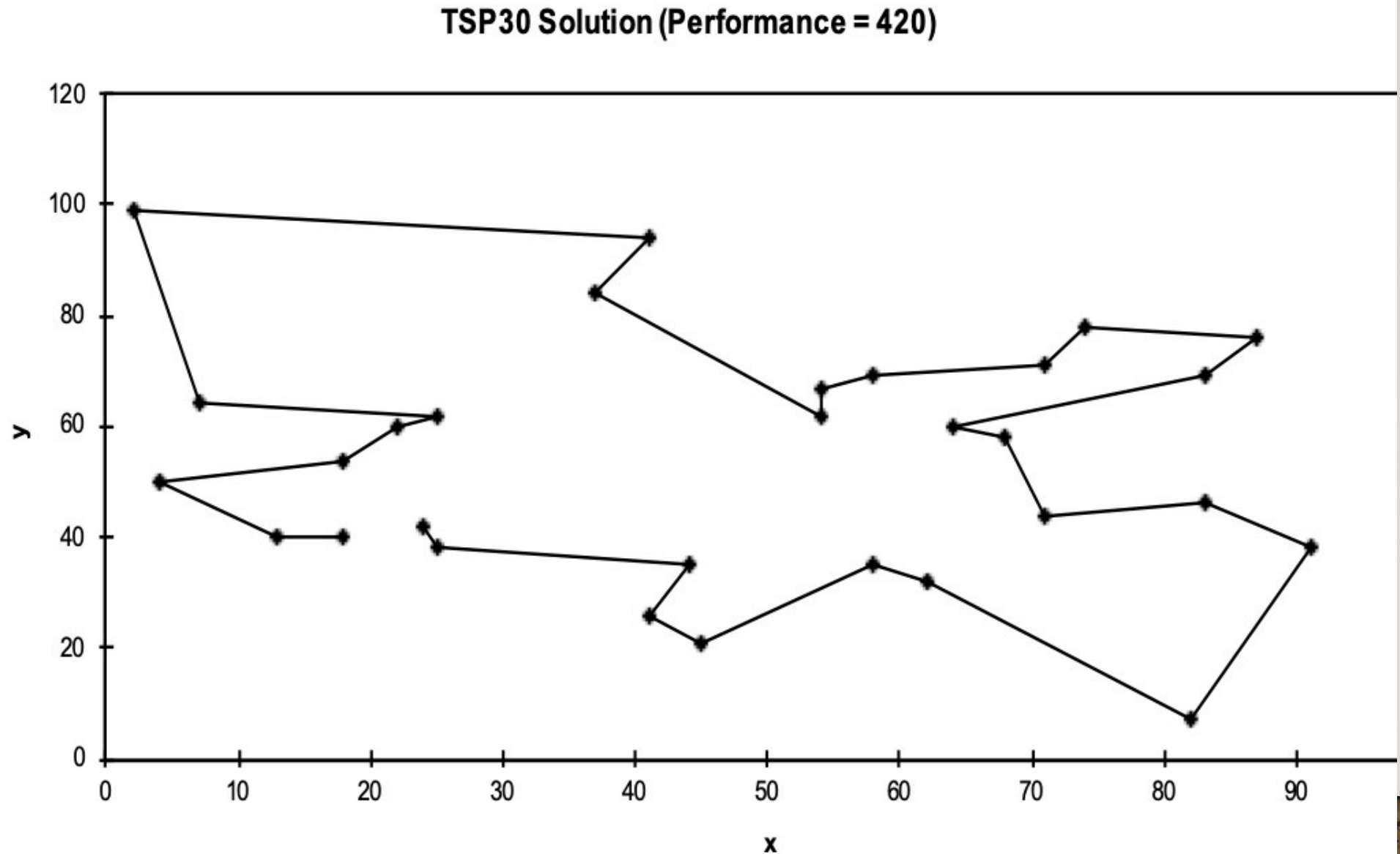
SOLUCION_j(DISTANCIA = 800)



SOLUCION κ (DISTANCIA = 652)

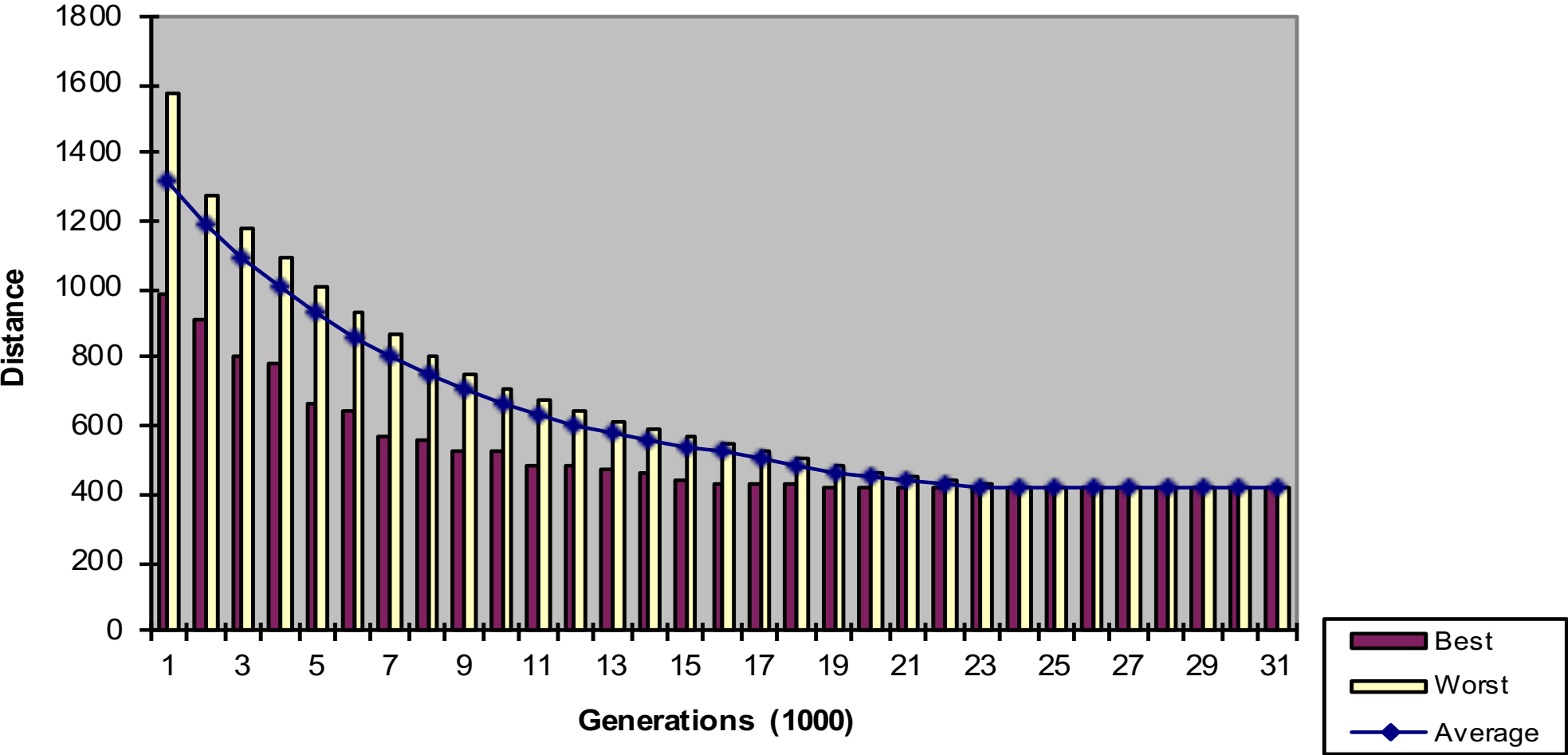


MEJOR SOLUCION (DISTANCIA = 420)



DE VERDAD VA MEJORANDO

TSP30 - Overview of Performance



OPERADORES GENÉTICOS

- Cruzamiento
 - PMX (partially mapped) Goldberg y Lingle
 - OX (order) Davis
 - CX (cycle) Oliver
- OX construye los hijos eligiendo una subsecuencia de un padre y preservando el orden relativo de las ciudades del otro padre. Las ciudades son copiadas en el mismo orden partiendo del segundo punto de corte, omitiendo los que ya aparecieron.

EJEMPLO OX

- Padres:
 - $p1 = (1\ 2\ 3\ | \ 4\ 5\ 6\ 7\ | \ 8\ 9)$
 - $p2 = (4\ 5\ 2\ | \ 1\ 8\ 7\ 6\ | \ 9\ 3)$

- primer paso:
 - $h1 = (x\ x\ x\ | \ 4\ 5\ 6\ 7\ | \ x\ x)$
 - $h2 = (x\ x\ x\ | \ 1\ 8\ 7\ 6\ | \ x\ x)$
- secuencia
 - $p2: \ 9\ 3\ 4\ 5\ 2\ | \ 8\ 7\ 6$
 - $p1: \ 8\ 9\ 1\ 2\ 3\ 4\ 5\ 6\ 7$
- segundo paso:
 - $h1 = (2\ 1\ 8\ | \ 4\ 5\ 6\ 7\ | \ 9\ 3)$
 - $h2 = (3\ 4\ 5\ | \ 1\ 8\ 7\ 6\ | \ 9\ 2)$

- Mutación

- intercambio del contenido de 2 genes elegidos aleatoriamente

- Ejemplo

| |

- $p = (2 \mid 8 \ 4 \ 5 \ 6 \ 7 \ 9 \ 3)$

- $h = (2 \mid 8 \ 4 \ 5 \ 3 \ 7 \ 9 \ 6)$

UN EJEMPLO NO MUY EXITOSO

Población: 4 individuos

Longitud: 8

P_{cruz} : 0.7

P_{mut} : 0.001

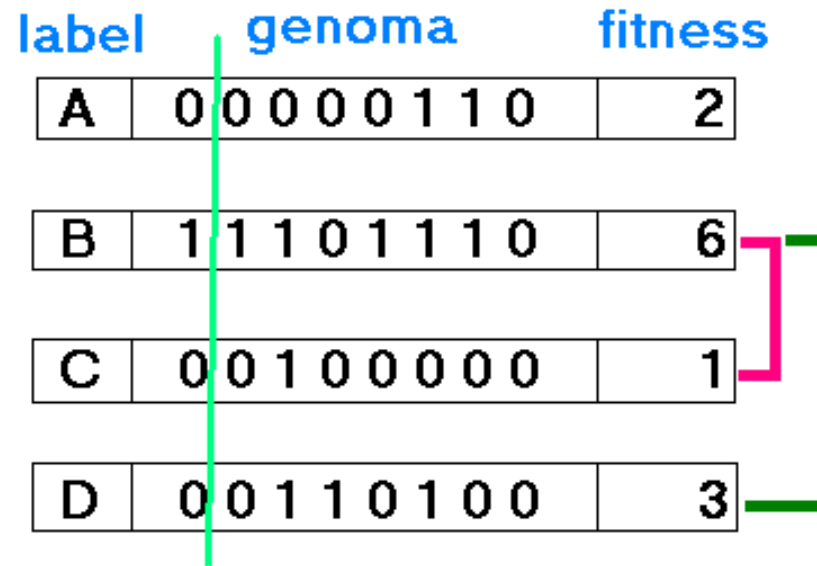
Fitness: cantidad de 1s del string

Valores típicos de tamaño de población y longitud del genoma

Están en el rango 50-1000

Población inicial (generada al azar):

label	genoma	fitness
A	00000110	2
B	11101110	6
C	00100000	1
D	00110100	3



Después del cruzamiento...

label	genoma	fitness
E	1 0 1 1 0 1 0 0	4
	★	
B	1 1 1 0 1 1 1 0	6
	★	
C	0 0 1 0 0 0 0 0	1
F	0 1 1 0 1 1 1 0	5

Después de la mutación:

label	genoma	fitness
E'	1 0 1 1 0 0 0 0	3
B'	0 1 1 0 1 1 1 0	5
C	0 0 1 0 0 0 0 0	1
F	0 1 1 0 1 1 1 0	5

- MUCHOS PROGRAMAS PUEDEN FORMULARSE PARA UN PROBLEMA DADO. ESTOS PROGRAMAS PUEDEN SER DIFERENTES EN MUCHOS ASPECTOS
-

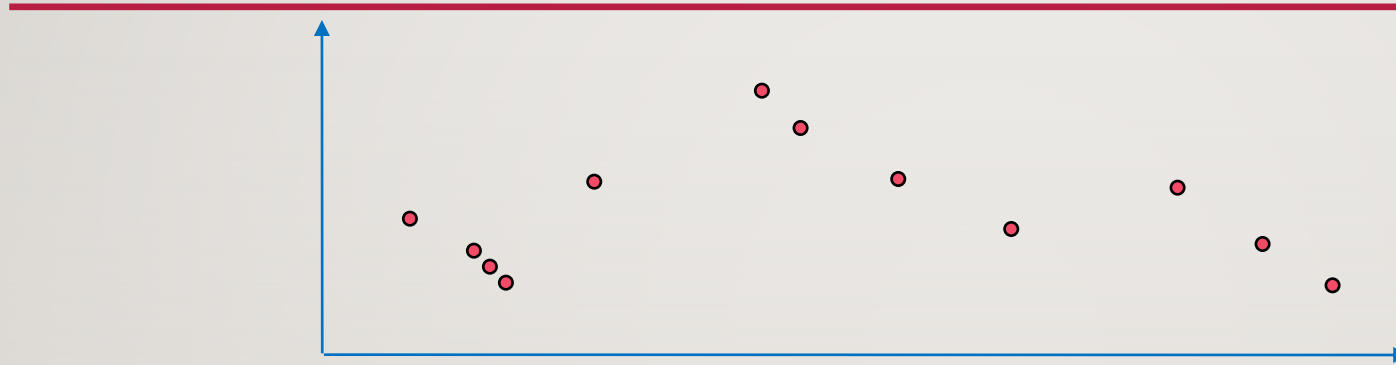
- Estructuras de datos utilizadas
- Operadores genéticos considerados
- Métodos para el manejo de restricciones
- Parámetros:
 - Tamaño de la población
 - Probabilidad de aplicación de operadores
 - etc.



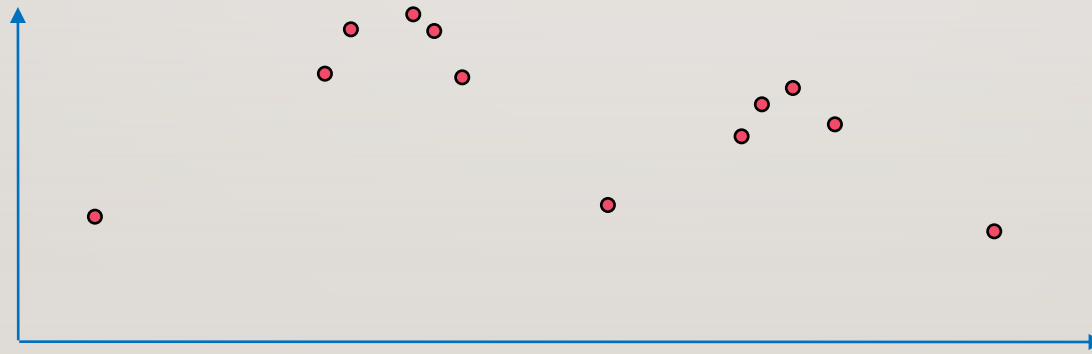
SIN EMBARGO, LA IDEA CENTRAL PERMANECE

- Una población de individuos sufre transformaciones, y durante este proceso de evolución, los individuos luchan por su supervivencia.

UN EJEMPLO ABSTRACTO



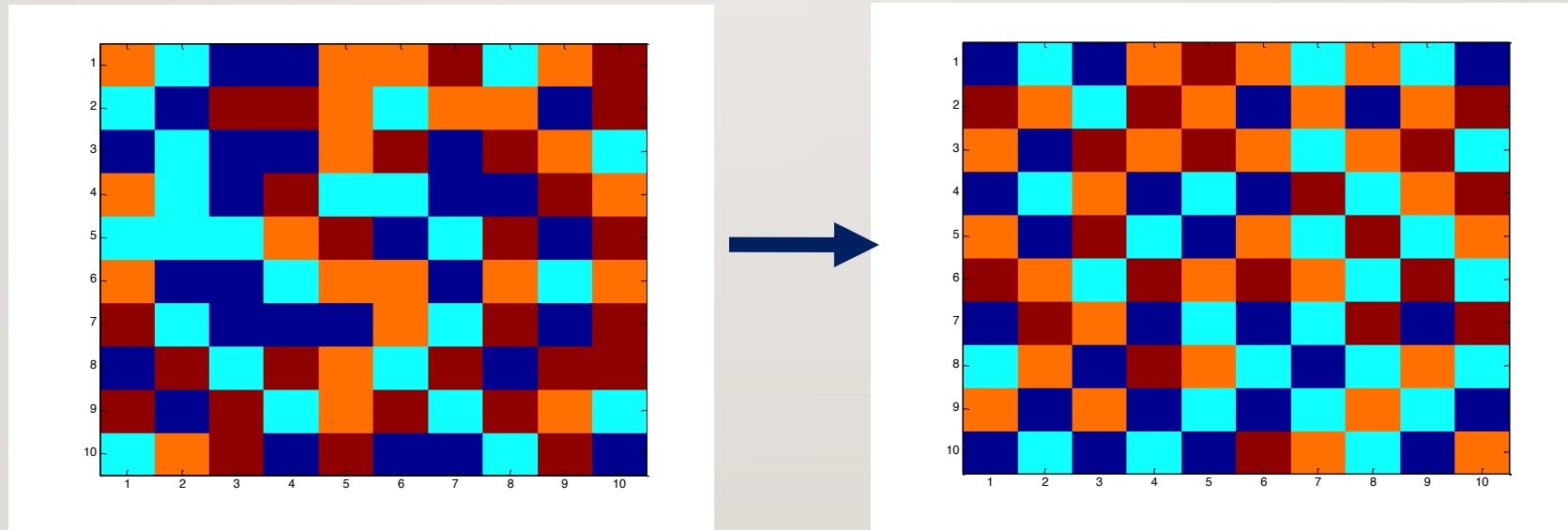
*Distribución de Individuos
en la Generación 0*



*Distribución de Individuos
en la Generación N*

Ejemplo tablero

- Tenemos un tablero de n por n espacios, en el cual cada espacio contiene un color elegido de un conjunto de cuatro colores diferentes.
- El objetivo es lograr un tablero tal que no haya vecinos con el mismo color



Ejemplo tablero

- ❖ Los cromosomas representan la manera en que el tablero está coloreado.
- ❖ Los cromosomas no están representados por strings de bits sino por una matriz de números
- ❖ Los número de la matriz pueden tener uno de cuatro valores: 0, 1, 2 o 3, dependiendo del color
- ❖ El cruzamiento, en este caso consiste en manipulación de una matriz. El cruzamiento será un intercambio de partes de la matriz, ya sea en forma horizontal, vertical, triangular o cuadrados.
- ❖ La mutación consiste en cambiar un número por otro.
- ❖ ¿Cuál sería el fitness?

ALGORITMOS EVOLUTIVOS

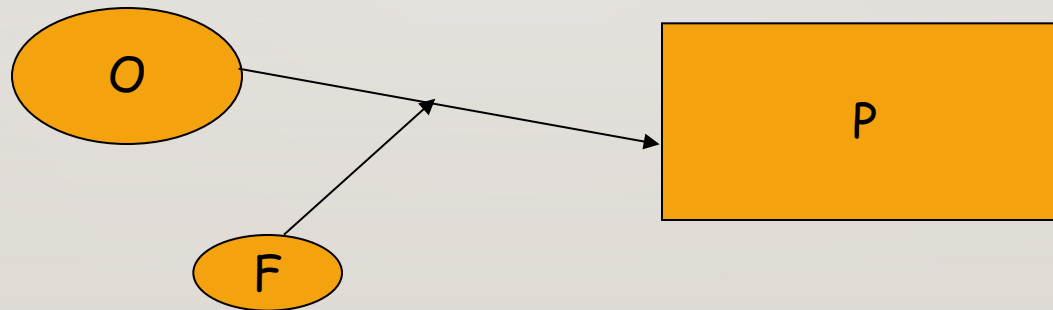


COMPUTACIÓN EVOLUTIVA

- ❖ Normalmente, la analogía es pobre entre:
 - ❖ Estructuras de datos y genotipos,
 - ❖ Soluciones y fenotipos
 - ❖ Operadores y mecanismos genéticos de transformación natural (mutación, recombinación, etc.)
 - ❖ *Fitness* usado de forma mediática en los procesos de selección.
- ❖ Más cercana a la replicación que a la selección *natural*
- ❖ Algoritmos genéticos, Estrategias evolutivas, Programación genética, Programación evolutiva ...

COMPUTACIÓN EVOLUTIVA

- ❖ Familia de métodos directos de búsqueda, estocásticos, basados en la población
- ❖ $P[t] = O[t] \times P[t-1]$
- ❖ P es una población de estructuras de datos que representan soluciones a un problema particular
- ❖ O es un conjunto de operadores de transformación usados para crear nuevas soluciones; F es una función de *fitness*



COMPUTACIÓN EVOLUTIVA

- ❖ ¿Es mágica? No.
- ❖ ¿Es universal? No. *Muy buena para ciertos problemas, muy mala para otros.*
- ❖ ¿Es fácil de aplicar? A veces ...
- ❖ ¿Por qué deberíamos interesarnos en CE? A veces se obtienen mejores resultados que con técnicas más tradicionales (no siempre). Buen marco general dentro del cual es posible crear algunos métodos de amplio poder para problemas específicos
- ❖ Usos: Optimización, problemas combinatoriales tales como Scheduling, Alife, Biología teórica

¿PARA QUÉ SIRVE?

- ❖ A veces, en combinación con otros métodos, es útil para:
 - ❖ Optimización variables complejas
 - ❖ Muchos problemas de optimización combinatorial
 - ❖ Mezcla de problemas de optimización discretos-continuos
 - ❖ Bases de evolución artificial
 - ❖ Diseño
 - ❖ Espacios de búsqueda de dimensiones desconocidas o variables

OPTIMIZACIÓN Y BÚSQUEDA

- ❖ Técnicas determinísticas clásicas (a menudo para problemas con variables continuas)
 - ❖ Métodos directos de búsqueda (sólo evaluaciones de funciones)
 - ❖ Métodos de gradiente descendente (hacen uso de información de gradiente, como en la regla delta)
- ❖ Operan en un espacio (completo o parcial) de soluciones posibles, saltando de una solución a la siguiente
 - ❖ Evaluativo
 - ❖ Heurístico
 - ❖ Estocástico

MÉTODOS DIRECTOS DE BÚSQUEDA

❖ ¿Cuándo se usan?

- ❖ La función a ser optimizada no es derivable, o está sujeta a errores random;
- ❖ Las derivadas de la función no son continuas, o su evaluación es muy cara y/o compleja.
- ❖ Falta de tiempo para aplicar métodos computacionales más costosos basados en gradiente.
- ❖ Puede que se requieran soluciones aproximadas en cualquier momento del proceso de optimización (los métodos directos trabajan refinando iterativamente la solución).

SIMULATED ANNEALING



- *Inspirado en el annealing de metales*

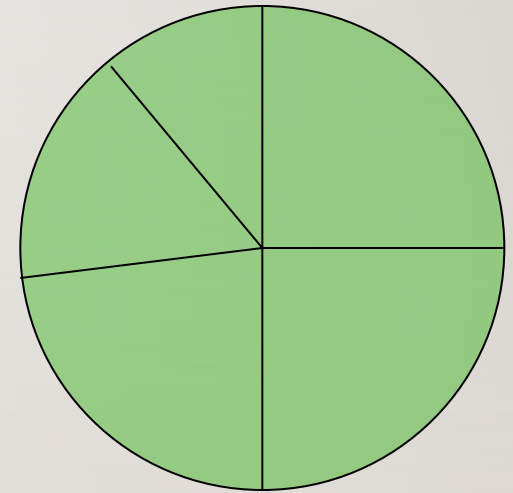
- 1) Inicialice T (análogo a la temperatura), genere una solución inicial, S_c , el costo de esta solución es C_c
- 2) Use un operador para generar de forma random una nueva solución S_n a partir de S_c , con costo C_n
- 3) Si $(C_n - C_c) < 0$, i.e. Se encontró solución mejor, hacer $S_c = S_n$. En caso contrario $\text{costo}[-(C_n - C_c)/T] > \text{random}$, entonces $S_c = S_n$, se acepta el movimiento con probabilidad proporcional a $\exp[-(C_n - C_c)/T]$.
- 4) Reduzca T , en general linealmente con el número de iteraciones
- 5) Si se alcanza el criterio de detención, fin; en caso contrario ir a paso (2)

FORTALEZA DE LOS AES

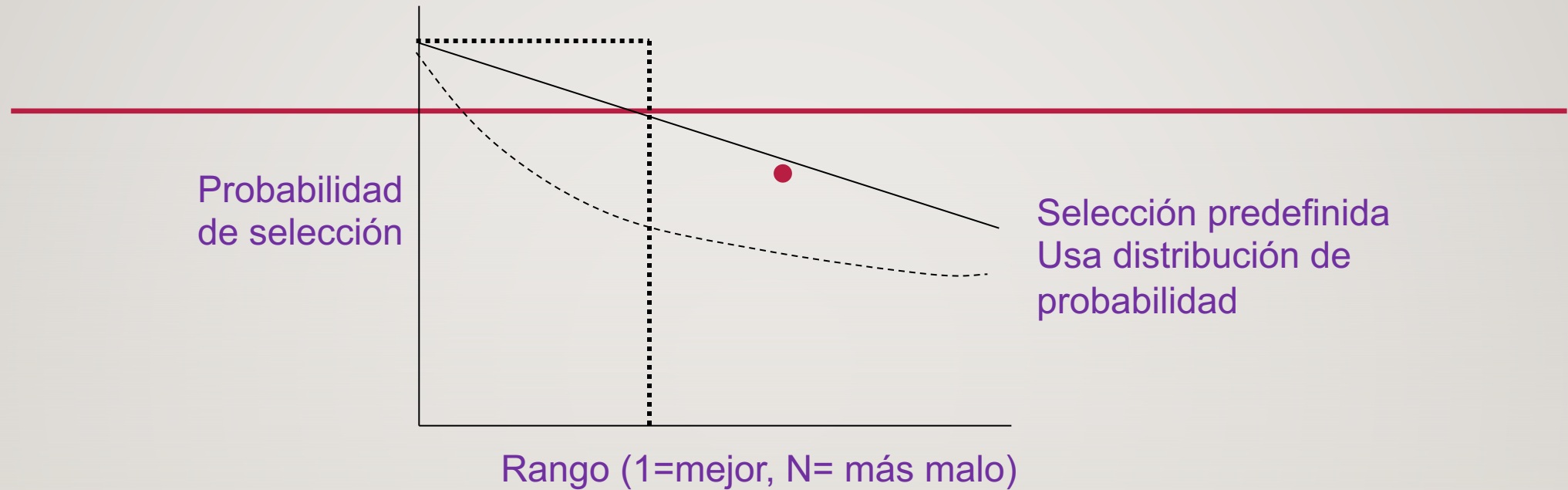
- ❖ En algún sentido, AEs atacan los problemas desde una perspectiva global, en vez de usar perspectivas puramente locales.
- ❖ Ya que se basan en la población, si se tratan adecuadamente se pueden explorar múltiples áreas del espacio de búsqueda en paralelo.
- ❖ Los elementos estocásticos en los algoritmos implican que no están necesariamente forzados a encontrar el óptimo local más cercano (como ocurre con los algoritmos determinísticos de búsqueda local).

SELECCIÓN: RULETA

```
for(i=0;i<POPSIZE;i++)  
    sum += fitness[i];  
  
for(i=0;i<POPSIZE;i++){  
    n=random(sum); ← rand num 0-sum  
    sum2=0;  
    i=0;  
    while(sum2<n){  
        i=i+1;  
        sum2=sum2+fitness[i];  
    }  
    Select(i);  
}
```



SELECCIÓN BASADA EN RANKING



Se hace un ranking en la población de acuerdo al fitness y luego se hace la selección de acuerdo a la distribución de probabilidad.

El truncado es un caso extremo.

Elitismo -> el mejor es elegido con probabilidad 1

SELECCIÓN POR TORNEO

- ❖ Se elige 2 miembros de la población, al azar, Padre1 = el mejor de ellos.
- ❖ Se elige 2 miembros de la población, al azar, Padre2 = el mejor de ellos.



ALGORITMOS DE ESTADO ESTACIONARIO

- ❖ Un miembro de la población cambia en el tiempo t , en vez de la generación completa
- ❖ Población inicializada al azar
- ❖ Se hace ranking (orden) de la población por *fitness*
- ❖ * Se elige pares de padres usando selección basada en ranking
- ❖ Reproducción para generar descendientes
- ❖ Agregar descendientes en la posición correcta (ordenados) en la población (sin repeticiones)
- ❖ Los últimos del ranking se van al infierno...
- ❖ Ir a *, a menos que se cumpla el criterio de detención

ASPECTOS GENERALES A TENER EN CUENTA

- Elección de cuestiones básicas de implementación:
 - Representación
 - Tamaño de la población, porcentaje de mutación, ...
 - Políticas de selección y eliminación
 - crossover, operadores de mutación
- Criterio de término
- Rendimiento, escalabilidad
- Una solución es sólo tan buena como la función de evaluación (la parte más difícil)

BENEFICIOS DE LOS ALGORITMOS GENETICOS

- Conceptos centrales son fáciles de entender
- Modulares, se pueden separar de la aplicación
- Permiten resolver problemas de optimización multi-objetivo
- Buenos para ambientes “ruidosos”
- Siempre entregan una respuesta; la que habitualmente mejora en el tiempo
- Inherentemente paralelos; se pueden trabajar de forma distribuida

BENEFICIOS DE LOS ALGORITMOS GENÉTICOS (CONT.)

- Se pueden mejorar las aplicaciones basadas en AGs a medida que se incrementa la experiencia en el dominio del problema
- Es fácil reutilizar aplicaciones previas (o alternativas)
- Es posible construir bloques básicos para aplicaciones híbridas
- Existe bastante historia acumulada sobre sus potenciales usos y aplicaciones

CUANDO USAR UN AG

- Cuando las soluciones alternativas son muy lentas o muy complicadas
- Se requiere una herramienta exploratoria para analizar casos nuevos
- Cuando existen problemas similares que ya han sido resueltos usando AGs
- Para hibridizar; es decir para combinar con una solución existente que usa otras técnicas
- Cuando los beneficios de los AGs encajan bien con el problema

TIPOS DE APLICACIONES TÍPICAS CON AG

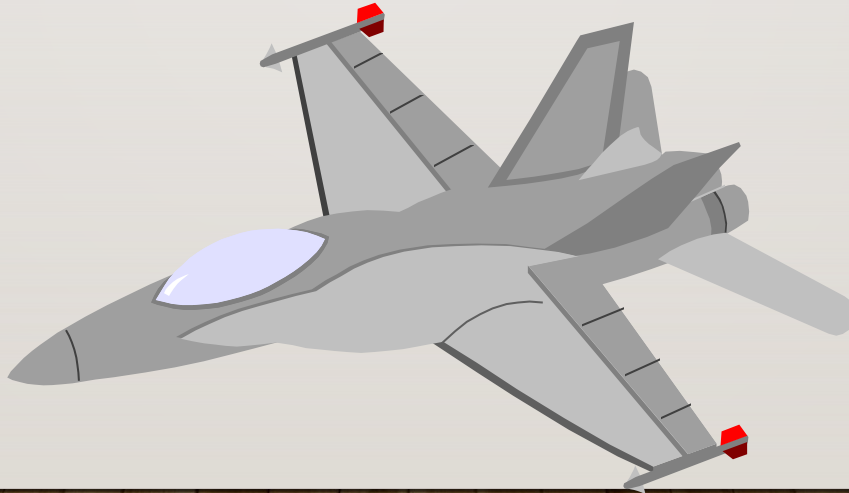
DOMINIO	APLICACIONES
Control	Evasión de misiles, balance de carga, gasoductos...
Diseño	Layout de circuitos, diseño de aeronaves, redes de comunicación
Scheduling	Industria manufactura, asignación de recursos
Robótica	Planificación de trayectorias
Machine learning	Diseño de redes neuronales, sistemas de clasificación
Procesamiento de señales	Diseño de filtros
Juegos	Póker, damas, ajedrez, dilema del prisionero
Optimización combinatoria	Set covering, diseño de rutas, coloración de grafos

COLORACION DE GRAFOS



EL PROBLEMA DE LOS VUELOS

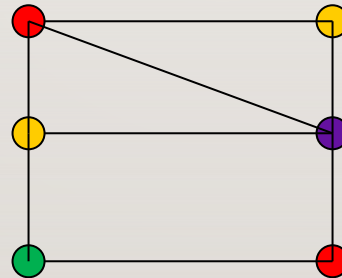
- Dado un conjunto de aviones que llegan a diferentes momentos, ¿cuántas mangas diferentes se necesitan para atenderlos a todos?



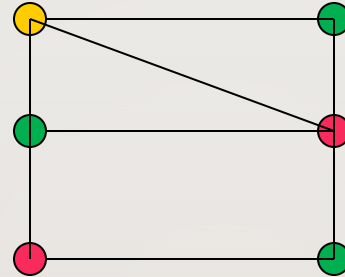
DEFINICIÓN

- Por coloración de vértices de un grafo G , se entiende la asignación de colores a los vértices de G , un color a cada vértice, de forma tal que vértices adyacentes tienen colores diferentes.

- Ejemplo:



○ también:



¿Cuál es el mínimo de colores necesarios?

OBS. Se trabajará sólo con grafos simples, ya que si hay un loop no hay coloración posible.

COLORACIÓN EN GRAFOS

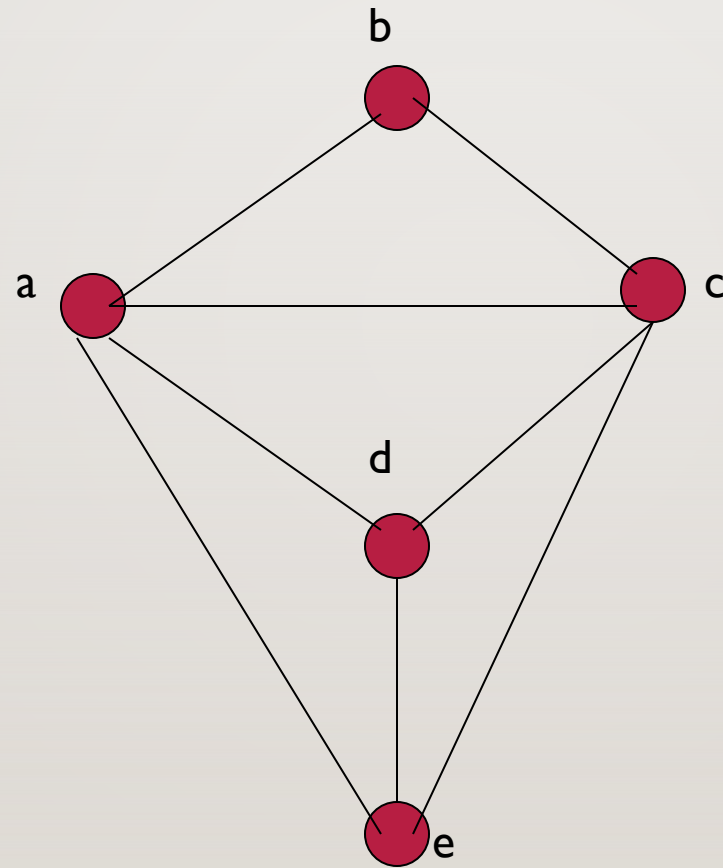
- Si $V(G)$ es un conjunto de cursos de la universidad, con aristas entre los cursos que tienen estudiantes en común, el número cromático del grafo es el mínimo de unidades de tiempo necesario para programar las clases sin que haya conflictos.

DEFINICIÓN

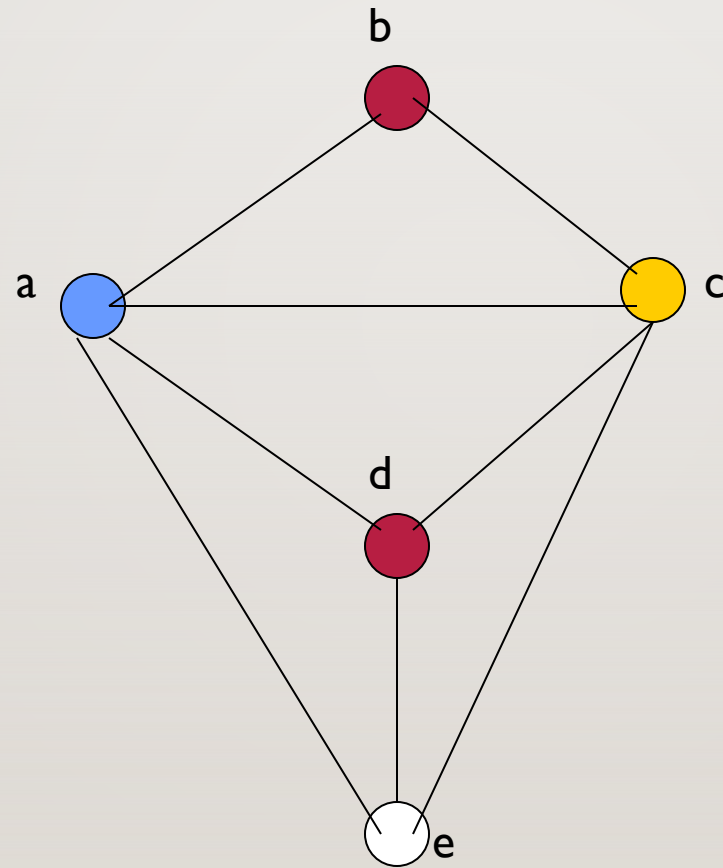
- Una k -coloración de G es un rotulado $f: V(G) \rightarrow \{1, \dots, k\}$.
Un grafo G es k -coloreable si tiene la propiedad de una k -coloración. El número cromático $\chi(G)$ es el mínimo k tal que G es k -coloreable. Si $\chi(G) = k$, pero $\chi(H) < k$ para cada subgrafo propio H de G , entonces G es k -crítico.

EJEMPLO

$\aleph(G) = ?$



EJEMPLO



$$\aleph(G) = 4$$

PROXIMA SEMANA

- Swarm Intelligence
- Colonias de hormigas